Introduction to IPv6

- IPv6 is the successor to IPv4 and is starting to take over networks worldwide
- IPv6 brings multiple improvements over IPv4, but there is one main reason for the switch
- Covered in CCNA exam topics:
  - **Topic 1.8:** Configure and verify IPv6 addressing and prefixes
  - **Topic 1.9:** Compare various IPv6 address types
- Also need to configure and verify the same kinds of static routes in IPv6 as already covered in IPv4

Lecture Overview

- **Hexadecimal review:** Previously covered with MAC addresses; important because IPv6 addresses are written in hexadecimal
- **Why IPv6 is necessary:** Overview of why we are moving away from IPv4
- **IPv6 basics:** Basic overview of IPv6, specifically IPv6 addresses
- **Configuration:** How to configure IPv6 addresses on Cisco devices
- **Bonus:** Practice question from Boson ExSim at the end of the lecture

---

Why Not IPv5?

- Internet Stream Protocol was developed in the late 1970s but was never introduced for public use
- It was never officially called IPv5, but it used a value of 5 in the Version field of the IP header
- IPv4 uses a value of 4 in the Version field; Internet Stream Protocol used a value of 5

- To avoid confusion, the successor to IPv4 was named IPv6 (uses a value of 6 in the Version field)

Hexadecimal Review

- Three numbering systems to know for CCNA:
    - **Binary (base 2):** prefix `0b`, available digits: 0 and 1
    - **Decimal (base 10):** prefix `0d`, available digits: 0–9
    - **Hexadecimal (base 16):** prefix `0x`, available digits: 0–9 and A, B, C, D, E, F

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9

Decimal	Binary	Hexadecimal
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

- **Example:** The number `10` is ambiguous without context
    - Could be decimal 10, binary `1 0` (decimal 2), or hexadecimal `10` (decimal 16)
    - Using prefix `0b` makes it clear it is a binary number

- Chart comparison (0 to decimal 15):
    - Up to 9, decimal and hexadecimal are the same
    - Decimal 10 = Hexadecimal A
    - Decimal 11 = Hexadecimal B
    - Decimal 12 = Hexadecimal C
    - Decimal 13 = Hexadecimal D
    - Decimal 14 = Hexadecimal E
    - Decimal 15 = Hexadecimal F
- Each hexadecimal digit contains **4 bits** of information
    - **Example:** Maximum value of four binary bits `1111` equals maximum hexadecimal digit `F`
- Decimal to Hexadecimal conversions to memorize:
    - 10 = A, 11 = B, 12 = C, 13 = D, 14 = E, 15 = F
- If you can convert between decimal and hexadecimal up to 15, and between decimal and binary, you can convert between any of the three systems

---

Conversion Examples

The `b` in `0b` and the `x` in `0x` are simply the **first letters of the word for the number base**.

- `0b` → **b**inary (base 2)
- `0x` → he**x**adecimal (base 16)

The `0` at the beginning acts as a signal that a special prefix is coming, similar to how we might write `0` before an octal number in some languages, or the `+` in a phone number.

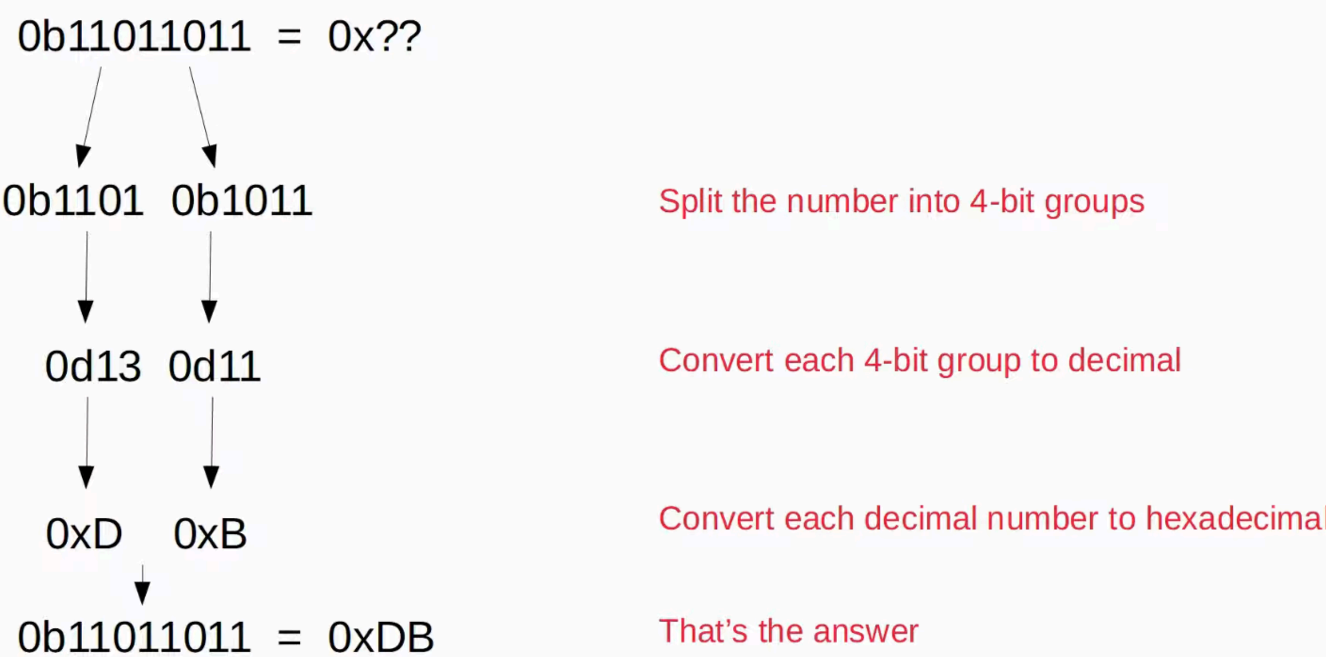
Why not `0h` for hexadecimal?

This is just a matter of convention that became standard in programming languages like C. `0x` was chosen, likely because "hex" starts with 'h' but `0h` might be confused with other things, and `x` is a distinctive, unambiguous letter associated with "hex." It's an arbitrary but universally accepted standard.

So, it "turns to" `0x` simply because the prefix for hexadecimal was standardized to be the `x` from "he**x**adecimal," just as the prefix for binary is the `b` from "**b**inary."

Binary to Hexadecimal

- Steps: Split into 4-bit groups → Convert each group to decimal → Convert decimal to hexadecimal → Combine hex digits
- **Example:** Binary `1101 1011` to hexadecimal



  - Split: `1101` and `1011`
  - `1101` = 8 + 4 + 1 = 13 → D
  - `1011` = 8 + 2 + 1 = 11 → B
  - Answer: Hexadecimal `DB`
  - Can verify using a calculator (e.g., Windows 10 programmer calculator mode)
- **Example:** Binary `0010 1111` to hexadecimal
  - Split: `0010` and `1111`
  - Convert each to decimal, then to hex
  - Answer: Hexadecimal `2F`

- **Example:** Binary `1000 0001` to hexadecimal
  - Split: `1000` and `0001`
  - Convert each to decimal, then to hex
  - Answer: Hexadecimal `81`

Hexadecimal to Binary

- Steps: Split hex digits → Convert each to decimal → Convert each decimal to binary
- **Example 1:** Hexadecimal `EC` to binary
  - E = 14 → `1110`
  - C = 12 → `1100`
  - Answer: Binary `1110 1100`
- **Example 2:** Hexadecimal `2B` to binary
  - 2 → `0010`
  - B (11) → `1011`
  - Answer: Binary `0010 1011`
- **Example:** Hexadecimal `D7` to binary
  - D (13) → `1101`
  - 7 → `0111`
  - Answer: Binary `1101 0111`

Why IPv6 Is Necessary

- The main reason for IPv6 is that there simply aren't enough IPv4 addresses available
- IPv4 is 32 bits → approximately 4,294,967,296 addresses (not enough for the modern world)
- When IPv4 was designed, the creators had no idea the Internet would be as large as it is today

- Techniques to preserve IPv4 address space (short-term solutions):
  - VLSM (Variable-Length Subnet Masks)
  - Private IPv4 addresses and NAT (Network Address Translation)
- These have been useful but are just short-term solutions
- The long-term solution is to transition to IPv6

IPv4 Address Assignment

- IPv4 address assignments are controlled by **IANA** (Internet Assigned Numbers Authority)
- IANA distributes IPv4 address space to **RIRs** (Regional Internet Registries), which assign them to companies (e.g., ISPs)
- RIRs:

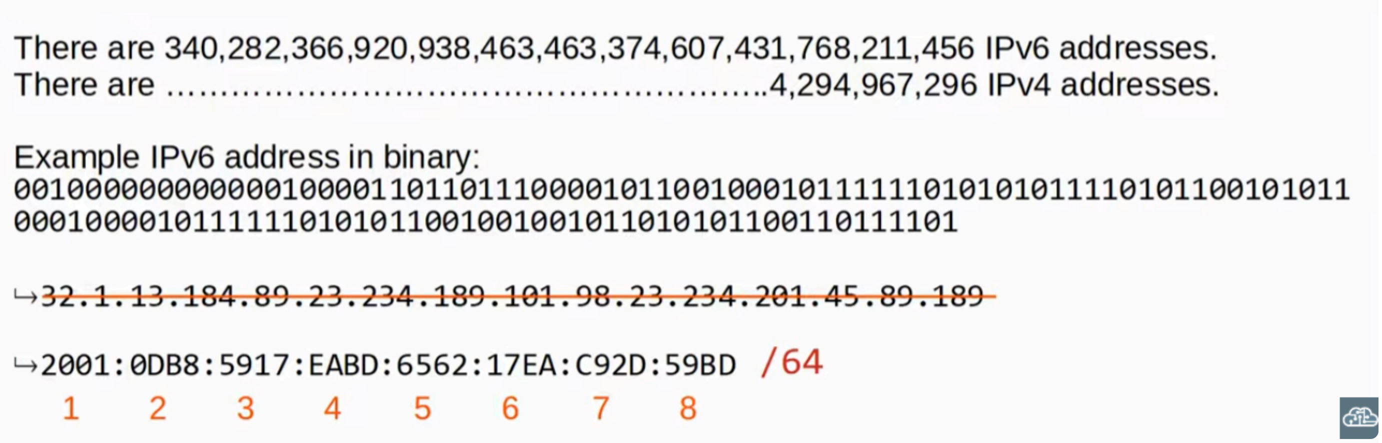
  - **AFRINIC** – Africa
  - **APNIC** – Asia-Pacific
  - **ARIN** – Canada, many Caribbean and North Atlantic islands, and the US
  - **LACNIC** – Latin America and the Caribbean
  - **RIPE NCC** – Europe, Middle East, parts of Central Asia

- RIRs are almost all out of IPv4 addresses
  - Example: September 2015 – ARIN declared exhaustion of IPv4 address pool
  - Example: August 2020 – LACNIC announced its final IPv4 allocation

---

IPv6 Basics

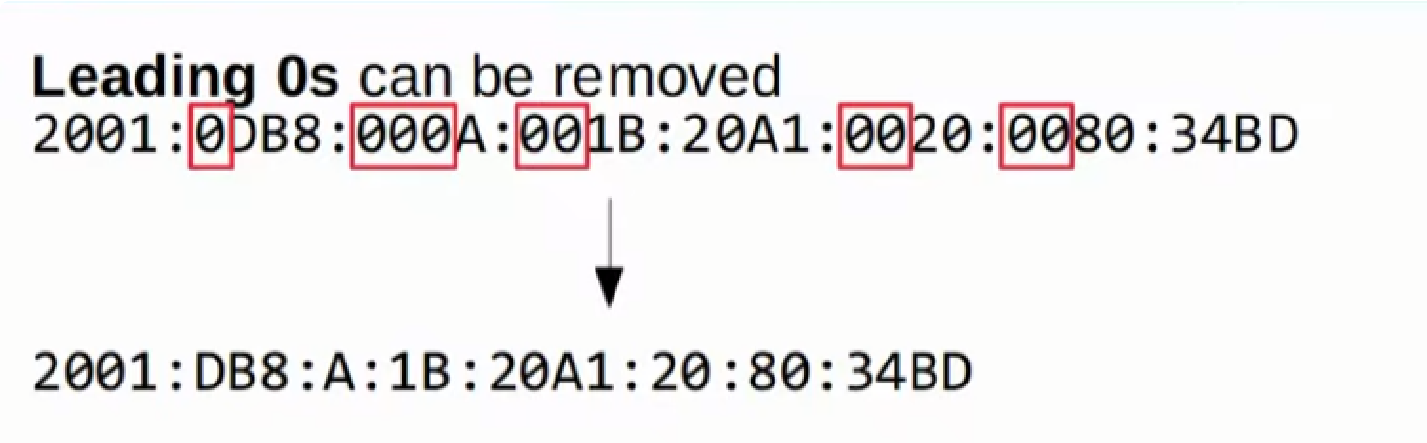
- An IPv6 address is **128 bits** long, which is 4 times the 32 bits of IPv4
- Every additional bit **doubles** the number of possible addresses



- IPv6 provides approximately $3.4 \times 10383.4 \times 1038$ unique addresses (you don't need to memorize the exact number)
- IPv6 addresses are not written in dotted decimal; they are written in **hexadecimal**
- Written as **32 hexadecimal characters**, divided into **8 groups of 4** using colons (these groups are called quartets)
- Uses **slash notation** (e.g., `/64`) to indicate the prefix length; no more dotted decimal subnet masks
  - Example: `/64` means the first half of the address is the network portion, and the second half is the host portion
- Methods exist to shorten IPv6 addresses to make them simpler to read

Shortening IPv6 Addresses

Rule 1: Remove Leading 0s



- Original IPv6:

  `2001:0DB8:000A:001B:20A1:0020:0080:34BD`

  Step 1: Remove leading zeros in each hextet.

  `2001:DB8:A:1B:20A1:20:80:34BD`

  Breakdown:

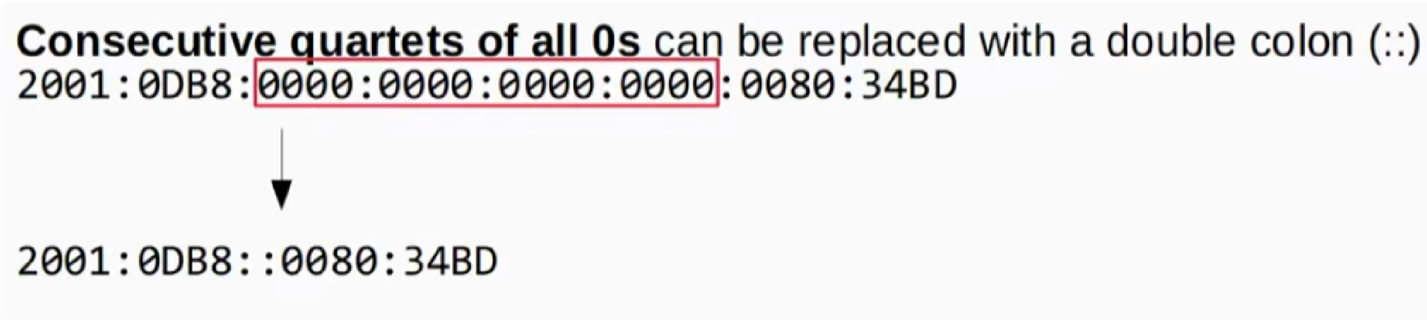
  `0DB8` → `DB8`
  `000A` → `A`
  `001B` → `1B`
  `0020` → `20`
  `0080` → `80`

  Final shortened IPv6:

  `2001:DB8:A:1B:20A1:20:80:34BD`

  Important: You **do not use `::` here** because there is no full `0000` hextet group to compress.

Rule 2: Replace Consecutive All-0 Quartets with Double Colon (::)



Given IPv6 address:

`2001:0DB8:0000:0000:0000:0000:0080:34BD`

Step 1: Identify consecutive all-0 hextets

Each hextet has 4 hexadecimal digits.

The double colon `::` can replace **any number** of consecutive all-zero quartets (hextets), from **1 up to 7**.

```
2001 : 0DB8 : 0000 : 0000 : 0000 : 0000 : 0080 : 34BD
                  ↑      ↑      ↑      ↑
          Four consecutive 0000 hextets
```

Step 2: Replace the consecutive `0000` hextets with `::`

```
2001:0DB8::0080:34BD
```

The `::` represents:

```
0000:0000:0000:0000
```

Step 3 (Optional): Remove leading zeros from the remaining hextets

- `0DB8` → `DB8`
- `0080` → `80`

Result:

```
2001:DB8::80:34BD
```

Final Shortened IPv6

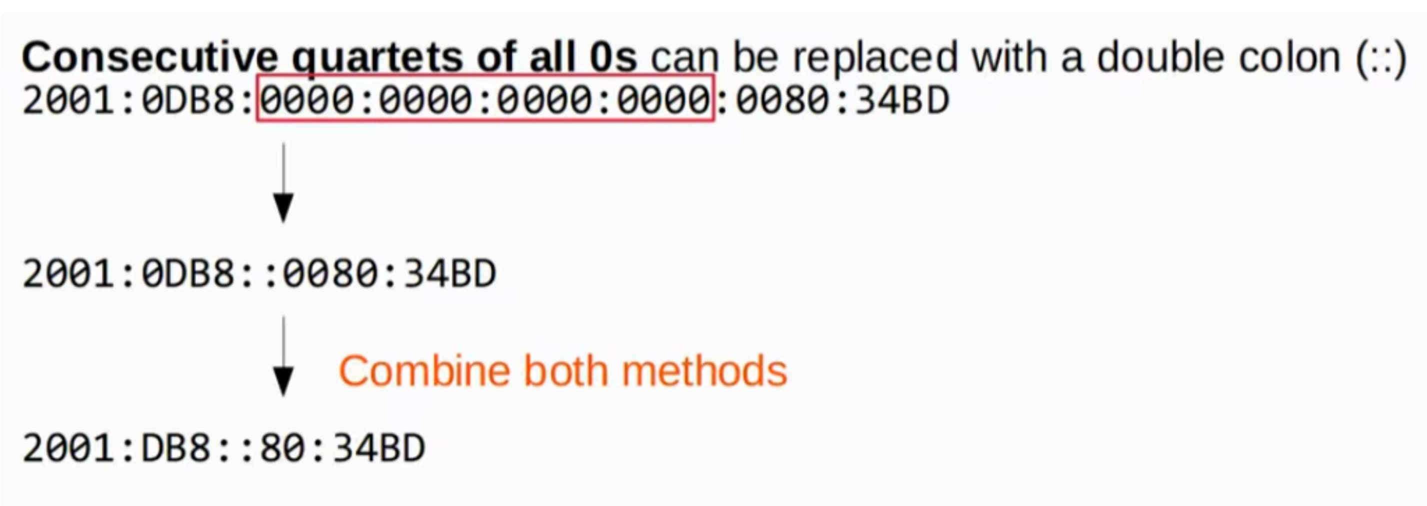
```
2001:DB8::80:34BD
```

Rules Used

1. Remove leading zeros from any hextet.
   - `0080` → `80`
   - `0DB8` → `DB8`

2. Replace one consecutive sequence of `0000` hextets with `::`.
   - Only **one** `::` is allowed in an IPv6 address, even if there are multiple groups of zeros.

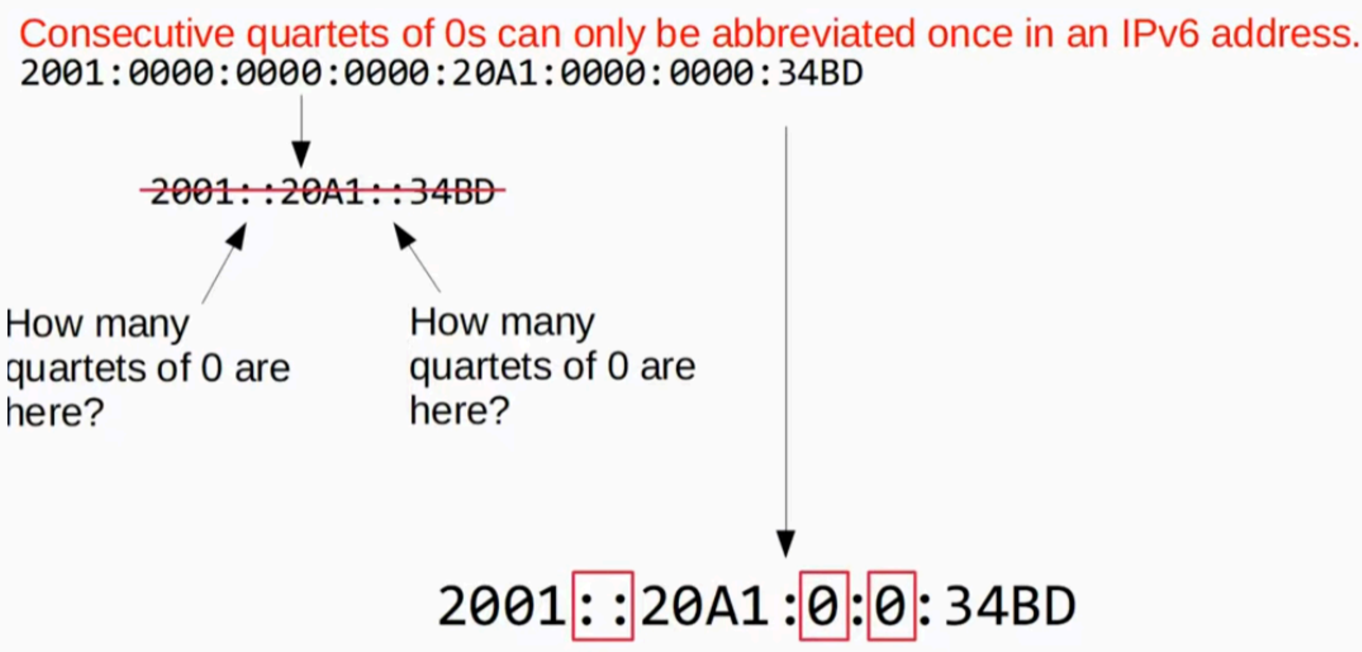
Combining Both Rules:



- You can remove leading 0s AND use the double colon

```
2001:0DB8:0000:0000:0000:0000:0080:34BD
       ↓ Remove leading zeros
2001:DB8:0:0:0:0:80:34BD
       ↓ Compress consecutive 0 hextets (::)
2001:DB8::80:34BD
```

- **Limitation: Double Colon Can Only Be Used Once**



  - If you use `::` twice, it becomes ambiguous how many all-0 quartets each represents

```
Original
2001:0000:0000:0000:20A1:0000:0000:34BD

✗ Invalid (uses :: twice)
2001::20A1::34BD

✓ Valid (compress only one sequence)
2001::20A1:0:0:34BD
```

Shortening IPv6 Practice

Full IPv6 Address	Shortened IPv6 Address
2000:AB78:0020:01BF:ED89:0000:0000:0001	2000:AB78:20:1BF:ED89::1
FE80:0000:0000:0000:0002:0000:0000:FBE8	FE80::2:0:0:FBE8
AE89:2100:01AC:00F0:0000:0000:0000:020F	AE89:2100:1AC:F0::20F
2001:0DB8:8B00:1000:0002:0BC0:0D07:0099	2001:DB8:8B00:1000:2:BC0:D07:99
2001:0DB8:0000:0000:0000:0000:0000:1000	2001:DB8::1000

- **Question 1:**
    - Remove leading 0s and use the double colon

```
1) Original
2000:AB78:0020:01BF:ED89:0000:0000:0001

↓ Remove leading zeros
2000:AB78:20:1BF:ED89:0:0:1

↓ Compress consecutive 0 hextets
2000:AB78:20:1BF:ED89::1
```

- **Question 2:**
    - Two sets of consecutive all-0 quartets, but can only use the double colon to replace one set

```
2) Original
FE80:0000:0000:0000:0002:0000:0000:FBE8

↓ Remove leading zeros
FE80:0:0:0:2:0:0:FBE8

↓ Compress the longest consecutive 0 hextets
FE80::2:0:0:FBE8
```

- **Question 3:**
  - Remove leading 0s and use the double colon

```
3) Original
AE89:2100:01AC:00F0:0000:0000:0000:020F

↓ Remove leading zeros
AE89:2100:1AC:F0:0:0:0:20F

↓ Compress consecutive 0 hextets
AE89:2100:1AC:F0::20F
```

- **Question 4:**
  - Remove leading 0s only; no consecutive quartets of all 0s to replace

```
4) Original
2001:0DB8:8B00:1000:0002:0BC0:0D07:0099

↓ Remove leading zeros
2001:DB8:8B00:1000:2:BC0:D07:99

↓ No consecutive all-0 hextets
2001:DB8:8B00:1000:2:BC0:D07:99
```

- **Question 5:**
    - Replace five quartets of all 0s with the double colon

```
5) Original
2001:0DB8:0000:0000:0000:0000:0000:1000

↓ Remove leading zeros
2001:DB8:0:0:0:0:0:1000

↓ Compress consecutive 0 hextets
2001:DB8::1000
```

Expanding Shortened IPv6 Addresses

- Steps to expand a shortened IPv6 address:
    - Add leading 0s back so each quartet has 4 hexadecimal characters
    - If a double colon `::` is used, replace it with all-0 quartets until there are 8 total quartets

- **Example:**

Put leading 0s where needed (all quartets should have 4 hexadecimal characters)

FE80::2:0:0:FBE8

FE80::0002:0000:0000:FBE8

If a double colon is used, replace it with all-0 quartets. Make sure there are 8 quartets in total.

FE80::0002:0000:0000:FBE8 5 quartets (8 quartets, but only 5 are written)

FE80:0000:0000:0000:0002:0000:0000:FBE8 8 quartets

```
 Shortened IPv6
FE80::2:0:0:FBE8

Step 1: Count written hextets
FE80 : 2 : 0 : 0 : FBE8
 1     2   3   4    5

Step 2: IPv6 needs 8 total hextets
8 − 5 = 3 missing hextets

Step 3: Replace :: with three 0000 hextets
FE80:0000:0000:0000:0002:0000:0000:FBE8

Final Expanded IPv6
FE80:0000:0000:0000:0002:0000:0000:FBE8
```

Expanding IPv6 Practice

Full IPv6 Address	Shortened IPv6 Address
FE80:0000:0000:0000:1010:02FC:0000:0009	FE80::1010:2FC:0:9
2001:0DB8:0001:0B23:2309:0000:0000:00C1	2001:DB8:1:B23:2309::C1
FD00:0000:0000:0000:1000:0689:9000:0CDF	FD00::1000:689:9000:CDF
FF02:0000:0000:0000:0000:0000:0000:0002	FF02::2
0000:0000:0000:0000:0000:0000:0000:0001	::1

- **Question 1 to 5:** Expand the provided shortened addresses into full IPv6 addresses by adding leading 0s and expanding the double colon

```
1) Full IPv6
FE80:0000:0000:0000:1010:02FC:0000:0009

Step 1: Remove leading zeros
FE80:0:0:0:1010:2FC:0:9

Step 2: Compress the longest 0 sequence
FE80::1010:2FC:0:9
```

```
2) Full IPv6
2001:0DB8:0001:0B23:2309:0000:0000:00C1

Step 1: Remove leading zeros
2001:DB8:1:B23:2309:0:0:C1

Step 2: Compress the longest 0 sequence
2001:DB8:1:B23:2309::C1
```

```
3) Full IPv6
FD00:0000:0000:0000:1000:0689:9000:0CDF

Step 1: Remove leading zeros
FD00:0:0:0:1000:689:9000:CDF

Step 2: Compress the longest 0 sequence
FD00::1000:689:9000:CDF
```

```
4) Full IPv6
FF02:0000:0000:0000:0000:0000:0000:0002

Step 1: Remove leading zeros
FF02:0:0:0:0:0:0:2

Step 2: Compress the longest 0 sequence
FF02::2
```

```
5) Full IPv6
0000:0000:0000:0000:0000:0000:0000:0001

Step 1: Remove leading zeros
0:0:0:0:0:0:0:1

Step 2: Compress the longest 0 sequence
::1
```

Finding the IPv6 Prefix (Network Address)

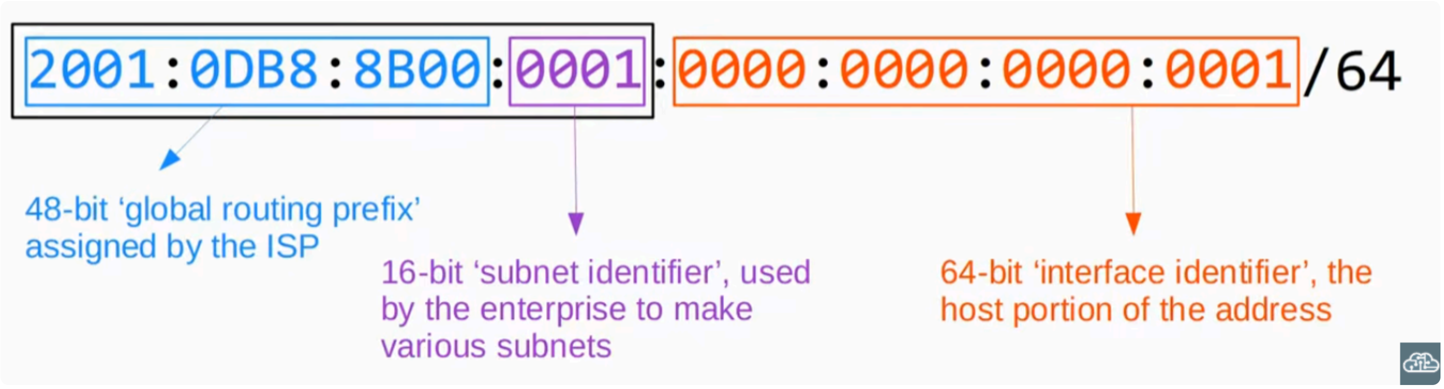
- Change all host bits to 0 to find the prefix

Typical Enterprise IPv6 Allocation

- Enterprise receives a **/48** block from ISP
- Typical IPv6 subnet uses **/64** prefix length
- This gives the enterprise **16 bits** for subnets (from bit 49 to bit 64)

- Remaining **64 bits** for hosts

IPv6 Global Unicast Address Structure



- **Global routing prefix:** First 48 bits (assigned by ISP) is in blue
  - **Purpose:** This portion uniquely identifies the specific network or site of the enterprise on the global internet.
- **Subnet identifier:** Next 16 bits (used by enterprise to create subnets) in purple
  - **Purpose:** This portion allows the enterprise to divide their assigned network into smaller internal sub-networks (subnets). Since they have 16 bits here, they can create up to 216216 (65,536) different subnets.
- **Host portion:** Last 64 bits is in orange
  - **Purpose:** This identifies the specific device (interface) within that specific subnet.
- Together, the global routing prefix and subnet identifier make the **network portion** (IPv6 network prefix)
- Note: This structure applies to IPv6 **global unicast** addresses — the regular addresses hosts can use over the Internet (not private or multicast addresses)

---

1. Finding the Prefix: /64 Example

- Finding the prefix with a /64 prefix length is easy: make the second half of the address all 0s

- **Example:**

2001:0DB8:8B00:0001:0000:0000:0000:0001/64

↓

2001:DB8:8B00:1::/64

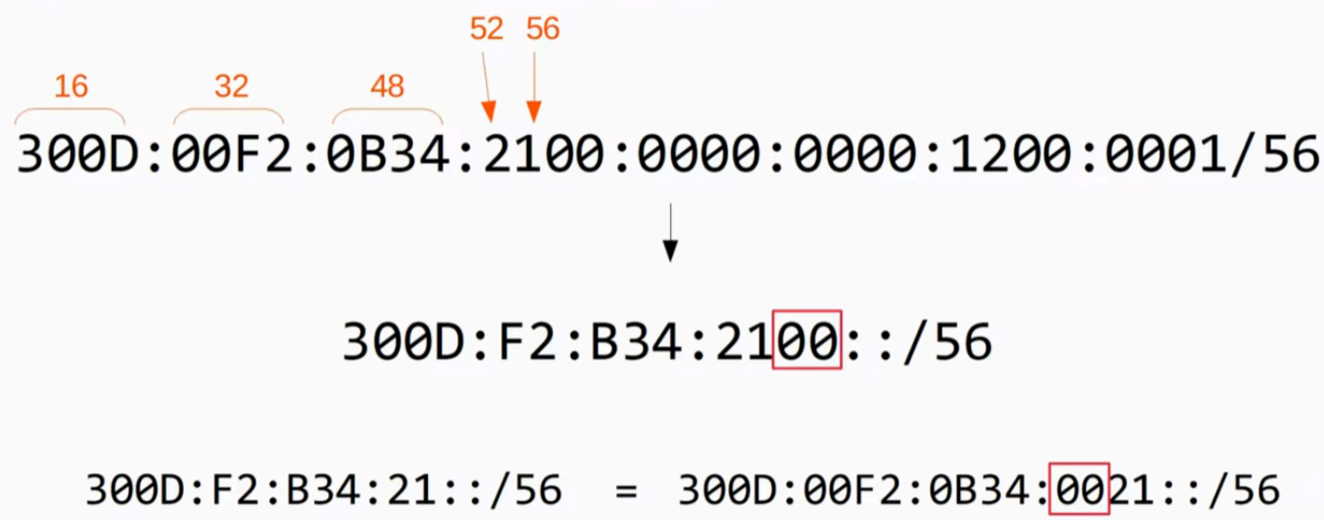
- Address: `2001:0DB8:8B00:0001:0000:0000:0000:0001/64`
- Host portion (last 64 bits) → change to all 0s
- Prefix `2001:DB8:8B00:1::/64` (shortened by removing leading 0 and using double colon for host portion)

```
Address: 2001:0DB8:8B00:0001:0000:0000:0000:0001/64

Step 1: /64 = first 4 quartets are network, last 4 are host

Step 2: Set host portion to all zeros
        2001:0DB8:8B00:0001:0000:0000:0000:0000

Step 3: Remove leading zeros
        2001:DB8:8B00:1:0:0:0:0

Step 4: Replace consecutive all-zero quartets with ::
        2001:DB8:8B00:1::

Prefix: 2001:DB8:8B00:1::/64
```

2. Finding the Prefix: Multiple of 4 Example

- If the prefix length is a multiple of 4, finding the prefix is straightforward because each hexadecimal character is 4 bits

- **Example: Address with /56 prefix length**

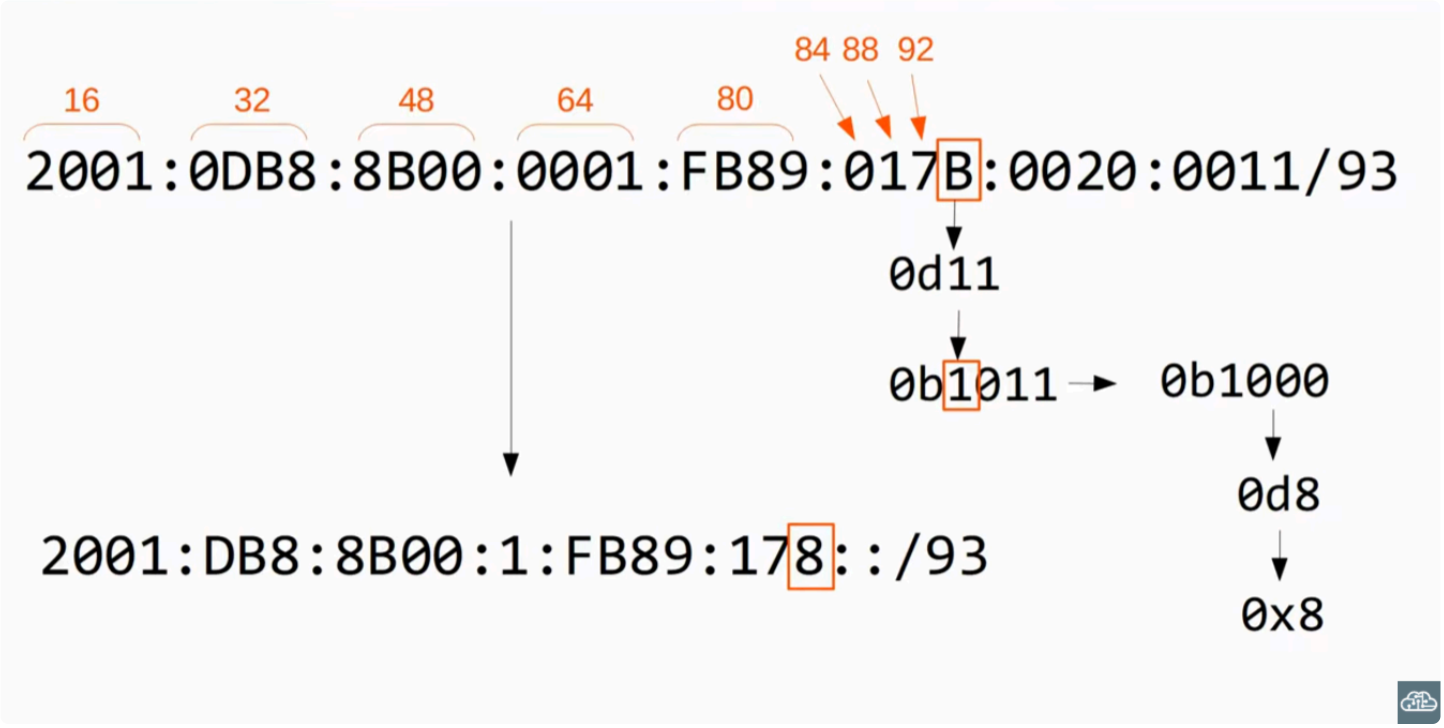


- Count bits through each quartet: 16 → 32 → 48 → 52 (in the 5th quartet) → 56 (in the 6th quartet)
- Network portion = first 14 hex characters (e.g., `2001:0DB8:1234:56`)
- Host portion → change to all 0s

```
Address: 300D:00F2:0B34:2100:0000:0000:1200:0001/56

Step 1: /56 = first 14 hex chars are network, rest are host
        Network: 300D:00F2:0B34:21
        Host:    00:0000:0000:1200:0001

Step 2: Set host portion to all zeros
        300D:00F2:0B34:2100:0000:0000:0000:0000

Step 3: Remove leading zeros
        300D:F2:B34:2100:0:0:0:0

Step 4: Replace consecutive all-zero quartets with ::
        300D:F2:B34:2100::

Prefix: 300D:F2:B34:2100::/56
```

Finding the Prefix: Not a Multiple of 4 Example

- If the prefix length is not a multiple of 4, the network portion ends in the middle of a hexadecimal digit, requiring binary conversion
- **Example: Address with /93 prefix length**



84 88 92
16
32
48
64
80
2001:0DB8:8B00:0001:FB89:017B:0020:0011/93
0d11
0b1011
0b1000
0d8
0x8
2001:DB8:8B00:1:FB89:178::/93

```
Address: 2001:0DB8:8B00:0001:FB89:017B:0020:0011/93

Step 1: Find where /93 falls
        Hextet 5 = 80 bits
        Hextet 6 = 96 bits
        Affected hextet: 017B

Step 2: Convert affected hextet to binary
        017B = 0000 0001 0111 1011

Step 3: Keep first 13 network bits (93 − 80 = 13)
        0000 0001 0111 1011
        Keep first 13: 0000 0001 0111 1
        Change host bits to 0: 0000 0001 0111 1000

Step 4: Convert back to hex
        0000 0001 0111 1000 = 0178

Step 5: Zero out remaining host hextets
        2001:0DB8:8B00:0001:FB89:0178:0000:0000

Step 6: Shorten
        Remove leading zeros and compress all−zero hextets
        2001:DB8:8B00:1:FB89:178::

Prefix: 2001:DB8:8B00:1:FB89:178::/93
```

Finding the IPv6 Prefix Practice

Host Address	Prefix
FE80:0000:0000:0000:4c2c:e2ed:6a89:2a27/9	FE80::/9
2001:0DB8:0001:0B23:BA89:0020:0000:00C1/64	2001:DB8:1:B23::/64
2001:0DB8:0BAD:CAFE:1300:0689:9000:0CDF/71	2001:DB8:BAD:CAFE:1200::/71
2001:0DB8:0000:FEED:0DAD:018F:6001:0DA3/62	2001:DB8:0:FEEC::/62
2001:0DB8:9BAD:BABE:0DE8:AB78:2301:0010/63	2001:DB8:9BAD:BABE::/63

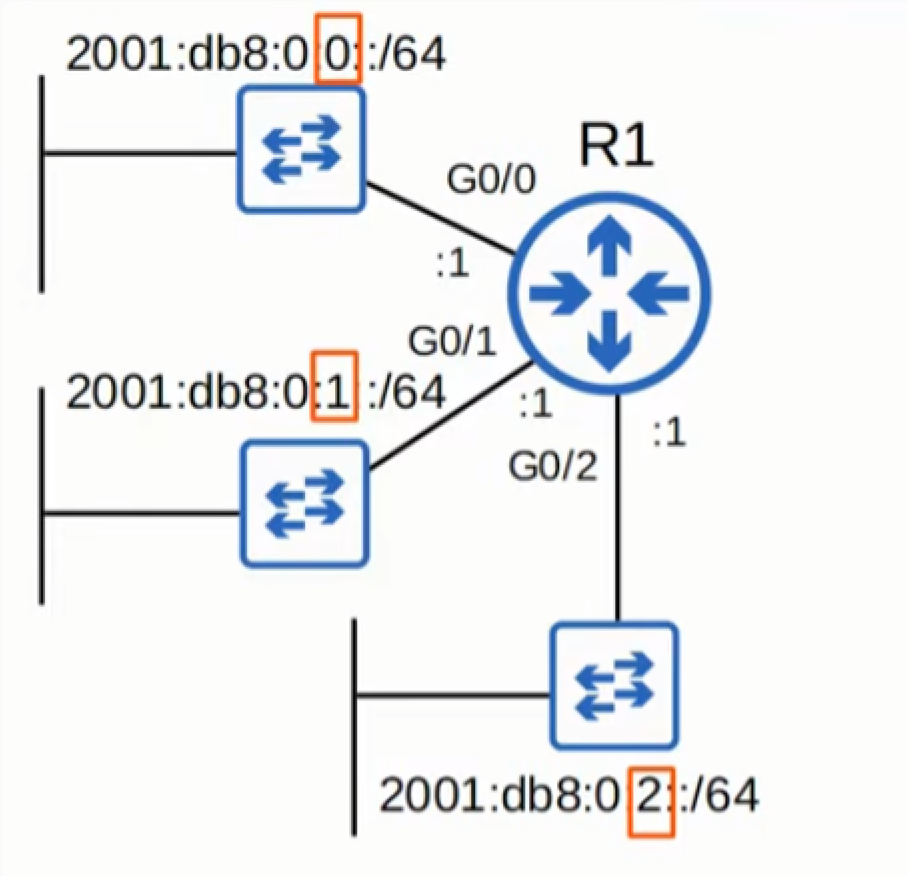
- **Practice:** Find the prefix of each provided IPv6 address
- You don't have to write out the shortened version; you can write out the whole address if you prefer
- For more practice, write out random IPv6 addresses with random prefix lengths and try to find the prefix of each

---

Basic IPv6 Configuration on Cisco Routers

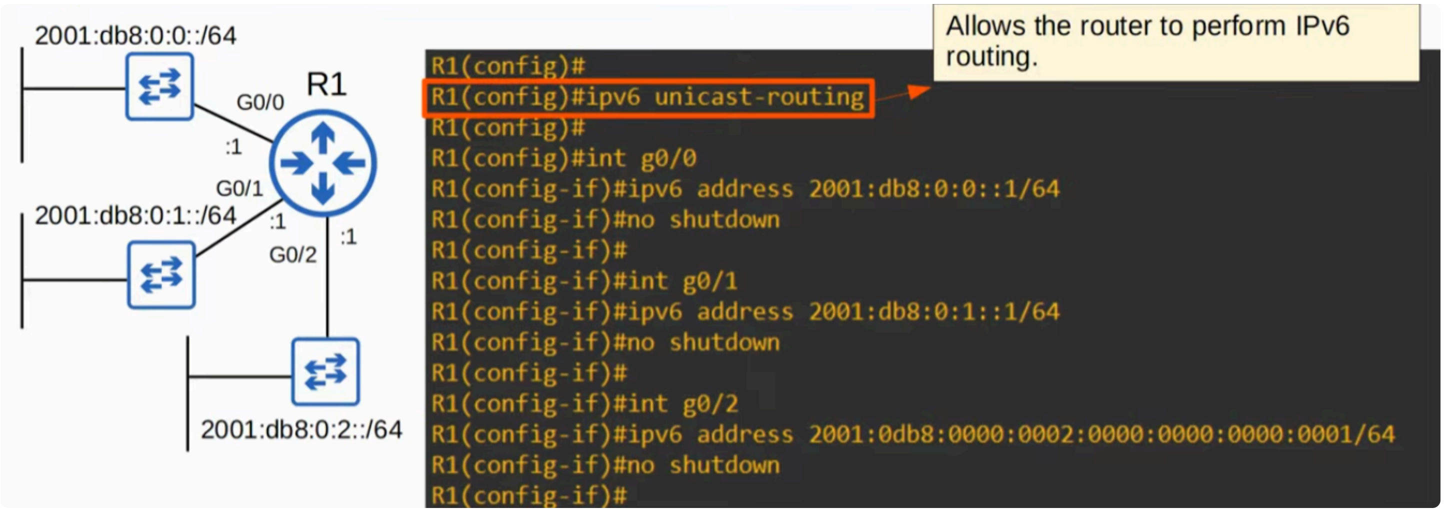
Topology Example

- R1 has three interfaces, each on a different subnet:



  - G0/0: `2001:db8:0:0::/64`
  - G0/1: `2001:db8:0:1::/64`
  - G0/2: `2001:db8:0:2::/64`
- Company assigned a `/48` block; using the last quartet of the prefix for subnets
- **Note:** `2001:db8` range is reserved for examples and documentation — should never be used in real networks

Enable IPv6 Routing




```
R1(config)# ipv6 unicast-routing
```

- **Note:** This command must be enabled or the router will not forward IPv6 packets

Configure IPv6 Addresses on Interfaces

```
R1(config)# interface GigabitEthernet0/0
R1(config-if)# ipv6 address 2001:db8:0:0::1/64
R1(config-if)# no shutdown
```

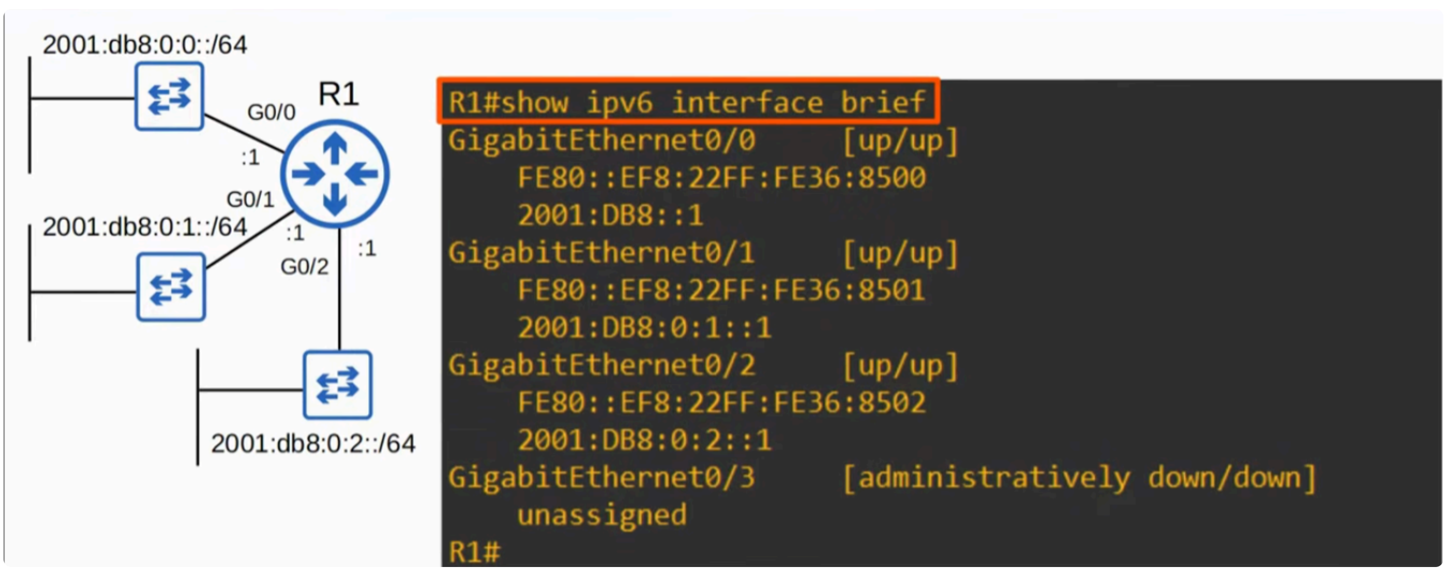
```
R1(config)# interface GigabitEthernet0/1
R1(config-if)# ipv6 address 2001:db8:0:1::1/64
R1(config-if)# no shutdown
```

```
R1(config)# interface GigabitEthernet0/2
R1(config-if)# ipv6 address 2001:0db8:0000:0002:0000:0000:0000:0001/64
R1(config-if)# no shutdown
```

- **Note:** You can use the full address, shortened address, or partially shortened address — the router understands all forms
- **Note:** Many IPv6 commands mirror IPv4 commands but use `ipv6` instead of `ip`

Verification

```
R1# show ipv6 interface brief
```



- Displays shortened versions of configured addresses
- Each interface shows **two** IPv6 addresses:
  1. The manually configured address
  2. A **link-local address** (automatically configured when an IPv6 address is assigned to the interface)

- Link-local addresses will be covered in detail in a later lecture (Day 32 – IPv6 address types)

Review

- Reviewed hexadecimal and practiced converting between it and binary
    - Even more important to be comfortable with hexadecimal for IPv6
- Introduced why IPv6 is necessary
    - Basically, there aren't enough IPv4 addresses for our modern world
- Covered the basics of IPv6, specifically IPv6 addresses
    - 128 bits in length, usually written using hexadecimal
- Showed basic commands to enable IPv6 on a router and configure IPv6 addresses on an interface